Improved AER Convolution Chip for Vision Processing with Higher Resolution and New Functionalities

Luis Alejandro Camuñas-Mesa, Alejandro Linares-Barranco, Antonio Acosta-Jiménez, Teresa Serrano-Gotarredona, and Bernabé Linares-Barranco

Instituto de Microelectrónica de Sevilla, IMSE-CNM, CSIC and Universidad de Sevilla. E-mail: luiscamu@imse.cnm.es

Abstract

We present a new neuromorphic fully digital convolution microchip for Address Event Representation (AER) spike-based processing system. This chip computes 2-D convolutions with a programmable kernel in real time. Previously, we designed and tested another convolution chip with a size of 32 x 32 pixels [1] and, based on the information obtained from this test, we have designed a new chip with larger resolution (64 x 64 pixels), improved behavior and new functionalities included. This chip receives and generates data in AER format, which is an asynchronous protocol, implementing the convolution of the input images with a programmable kernel. The most important new functionality included in this chip is the multikernel capability, which allows us to program several kernels (up to 32) so that each input event will be processed with the corresponding kernel, depending on the origin of the input event. The paper describes the architecture of the chip, with special emphasis to the new improvements.

1. Introduction

Conventional image processing systems operate on sequences of frames. An image is captured each frame period (which is 25-30 ms for commercial video cameras) and the processing algorithm is applied for each acquired frame. But biological brains do not work this way. In a biological vision system the information is acquired and processed continuously in time. Biological retinae send the image information to the visual cortex coded as spikes (also called events). When the activity level of a certain pixel reaches a threshold the pixel sends a spike. Very active pixels send more spikes, and spikes propagate through the processing chain as soon as they are generated without waiting for the whole image to be processed.

Consider, for example, the vision processing system shown in Fig. 1 where the image from the retina is processed by two layers of convolutions. If the system in Fig. 1 is frame-based, each layer of convolutions will have to wait until the previous layer finishes processing the whole image to start its operation. However, if information between chips is sent as asynchronous spikes, spikes are communicated and processed by the next layer as soon as they are generated.

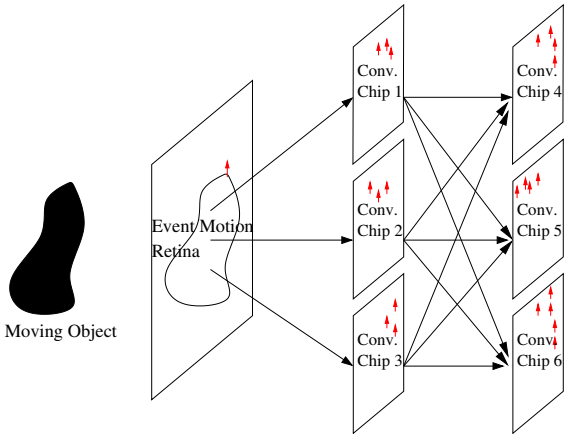

Fig. 1. Multilayer vision processing system

In fact, the system in Fig. 1 resembles the architecture of biological vision systems where the image from the retina is sent to the visual cortex. The visual cortex is organized as layers of neurons. Neurons from each layer project their output to a population of neurons of the following layer, thus performing a projection field or, equivalently, a convolution operation [2].

The Address-Event-Representation (AER) protocol allows asynchronous massive interconnection of neuromorphic processing chips. The AER protocol was first proposed in 1991 in

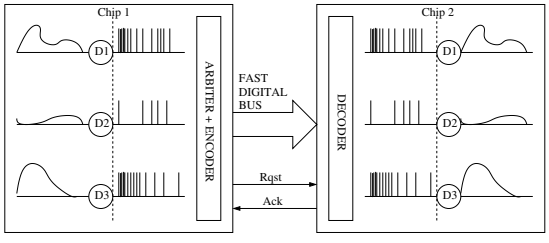

Fig. 2. Address Event Representation (AER) protocol

Caltech [3] to solve the problem of massive interconnectivity between populations of neurons located in different chips. Fig. 2 illustrates the interconnection of two chips through a point-to-point AER link. Assume that chip1 in Fig. 2 contains a matrix of M x N pixels that have to be interconnected with the pixels of chip2, which are also arranged as an M x N matrix of pixels. The AER protocol multiplexes all these connections through a high-speed digital bus. Each pixel in chip 1 (or sender chip) converts an input signal into a train of pulses whose frequency is proportional to its signal level. These pulses are arbitrated and the address of the sending pixel is coded on the fast digital bus asynchronously with handshaking. Chip 2 (or the receiver chip) decodes the arriving address and sends a spike to the corresponding neuron so that the original signal can be reconstructed.

As an additional advantage, the AER protocol allows to perform operations in the address space as addresses travel from chip to chip. For instance, by inserting a look-up table transforming the addresses in the digital bus, an image rotation or any other image transformation can be easily performed on the fly. Furthermore, if for each event sent to the receiver chip, an event is sent not only to its corresponding pixel but to all the pixels in its neighborhood, a projection field can be implemented. We additionally weight the events sent to the neighbor pixels with a stored kernel, such that the resulting image is the convolution of the input image with the stored kernel [4].

In this paper, we present a fully digital AER convolution chip with 64 x 64 pixels. In Section 2, we will comment the different convolution chips that we have designed previously, while in Sections 3 and 4 we will describe the improvements added to the new version, at the pixel and system levels, respectively. Finally, in Section 5 we will give some conclusions.

2. Architecture of the convolution chip

Before describing the chip presented in this paper, we will make a brief revision of our previous work in this field: a first convolution chip based on an analog pixel, a fully digital convolution chip with 32 x 32 pixels, and now the aim of this paper is a new fully digital convolution chip with 64 x 64 pixels. In the following sections we will compare this new chip with the previous version, the 32 x 32 convolution chip.

A. Mixed-signal convolution chip with analog pixel:

The main difference between the first convolution chip that we designed and the latest one was at the pixel level. The first convolution pixel was analog, it was based on a capacitor that integrates current pulses (that are proportional to the kernel values) when input events are received, until a threshold voltage is reached, producing an output spike [5], [6].

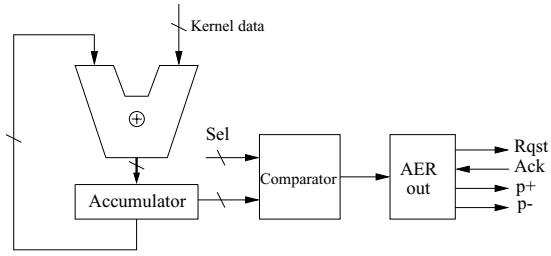

Fig. 3. Architecture of the digital pixel

This version of the convolution chip had some drawbacks, like the necessity of implementing calibration techniques for mismatch compensation, the large amount of bias signals that had to be set and the power consumption. These drawbacks motivated the design of a digital version of the pixel.

B. Digital convolution chip 32 x 32:

To overcome the main drawbacks of the analog pixel, a fully digital pixel was designed. It was based on a full adder that accumulates the kernel values when input events arrive until a programmable threshold is reached. The basic architecture of the digital pixel can be seen in Fig. 3.

Using this digital pixel, we designed and tested a 32 x 32 convolution chip, with satisfactory results [7]. With the help of the information obtained from this chip we found out that the number of bits in the pixel adder and in the kernel data could be reduced.

Also, we redesigned several blocks of the digital convolution chip in order to improve its basic behavior and to add some new functionalities that will be described in this paper.

C. Digital convolution chip 64 x 64:

First, we will describe the global system-level architecture of the convolution chip, and in the following sections the specific improvements over the previous version will be described in more detail.

Fig. 4 shows a scheme of the convolution chip, receiving an input address event and generating an output address event. We have the following blocks:

- Array of 64 x 64 digital pixels
- Static RAM that holds the stored kernel in 2's complement representation

- Synchronous controller, which performs the sequencing of all operations for each input event, which basically consists of adding row by row the kernel onto the array of pixels
- High-speed clock generator, used by the synchronous controller. It is possible to use either this internal clock or an external one.
- Configuration registers, that store several parameters loaded serially at startup. We will talk about these parameters when describing the multikernel operation.
- A block that computes the 2's complement of the RAM kernel data before sending them to the pixels. This block is necessary because the input events entering the chip are signed. If the incoming event has positive sign the weights are left unchanged. However, if the incoming event is negative the weights are inverted by the 2's complement block. This way, the sign multiplication is done in the periphery instead of complicating the pixel.
- Left/Right Column Shifter, which is necessary to center the kernel around the pixel indicated by the input address.
- AER-out, asynchronous circuitry for arbitrating and sending out the output address events generated by the array of pixels.

3. Improvements at the pixel level

For the new convolution chip, we have designed a new pixel that improves the previous one. The most important changes are: the reduction of the size of the adder, the reduction of the power consumption and

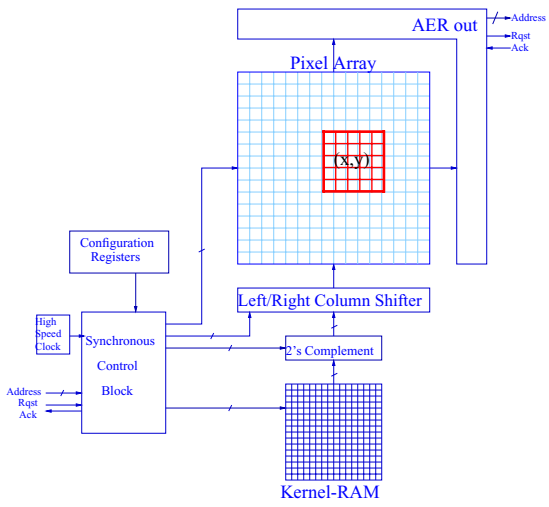

Fig. 4. Architecture of the convolution chip

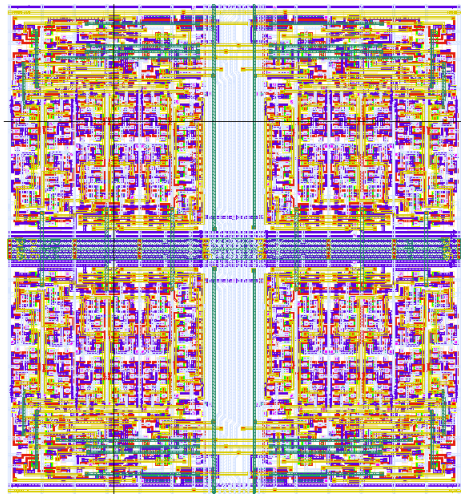

Fig. 5. Layout of the 2 x 2 cluster of pixels

the implementation of a new functionality, the positive or negative events inhibition.

A. Adder size reduction

In the first version of the digital pixel, we decided to use an 18-bit adder (17 bits plus the sign), with 6-bit data in the kernel. Also, we introduced the possibility of selecting the effective size of the adder between 8 different positions of the accumulator. The reason for this decision was to be able to discriminate between different integrator levels when using maximum-size kernels.

However, after testing the previous chip, we found out that for certain applications (where we want to obtain fast outputs of the convolution operation) it would be more interesting to have smaller accumulators. So, we decided to reduce it and design our new pixel with a resolution of 6 bits (5 plus the sign), and 4-bit data in the kernel. We also have reduced the programmability of the threshold, and now we only have two levels.

With these new specifications, we have designed a new pixel with the aim of placing 4 new pixels (a cluster of 2 x 2) in the same area occupied by one old pixel, so that we could build an array of 64 x 64 pixels with no cost in terms of area.

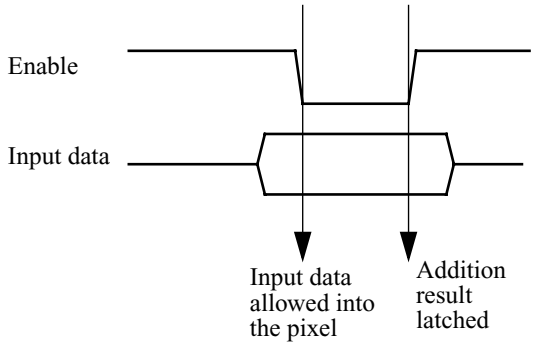

Fig. 6. Behavior of the pixel

Fig. 5 shows the layout of the 2 x 2 cluster of pixels, with an area of 115 μm x 107 μm.

B. Power consumption reduction

As was briefly described in the previous section, when an input event arrives the synchronous controller activates sequentially the corresponding rows of the kernel in order to be added to the selected rows of the array. So, when a row of the kernel is enabled, each bit of the row is connected to the adders in all the pixels of the same column of the array. This way, although only one row receives the order of saving the result of the addition, all the rows implement the operation without saving the result. This was the negative consequence of sharing each output of the RAM for all the pixels in the same column, and this situation was producing that all the pixels were consuming unnecessary power.

However, in the new version of the pixel we have used the same signal that enables the storage of the result of the addition to control a set of switches that prevents the data from being added needlessly. Fig. 6 shows the behavior of these switches.

C. Events inhibition

This is a new functionality for the chip that we have implemented at the pixel level. In some applications, it can be useful to inhibit the output events generated by the convolution operation depending on the sign of these events, because the information carried by either the positive or the negative events is not important.

It would be easy to discard these events once they are out of the chip, or even during the arbitration process inside the chip, but both of these possibilities would consume resources without any benefit. Thus, we have implemented this functionality inside the pixel, so that there is no communication between the pixel and the periphery when there is no output event to be generated.

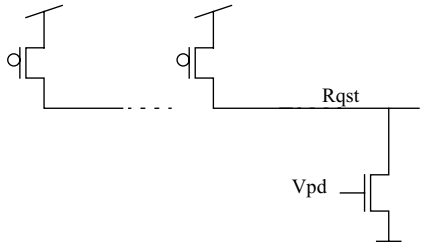

Fig. 7. Communication scheme in the previous version

There are two control bits (accessed from the parameter register) that enable the inhibition of either the positive or the negative events. Then, if the positive events are inhibited, once the accumulator reaches the positive threshold, the pixel resets itself, waits for the next event and nothing else happens.

4. Improvements at the system level

We also have included some other improvements to the convolution chip, like a more robust communication scheme between the pixels and the arbiter, a larger image resolution and the implementation of the multikernel capability.

A. Communication between pixels and arbiter

When one pixel reaches the selected threshold, it fires an event. This event will reach the rows arbiter, and once this arbiter acknowledges, the event is transmitted out of the chip. The request input of the arbiter is a wired-or of the request signals generated by all the pixels in the same row. The communication scheme used in the previous version is shown in Fig. 7.

This scheme was working properly, but we needed to adjust bias voltage Vpd, which was not very robust, specially when scaling up the number of pixels. The problem of this structure is that (1) we have to ensure that each pMOS transistor on its own will be able to activate the Rqst node against the pull-down transistor, and (2) the pull-down transistor must be able to reset the Rqst when all pMOS transistors are active. And both of these operations must be done with a limited delay.

To avoid these problems, we have implemented the new communication scheme shown in Fig. 8 [8]. Here, the pull-down transistor that resets the common node to all the pixels is controlled by the acknowledge signal generated by the arbiter, and not by a bias voltage as before. This way, this pull-down is not active when the pixels are triggering the Rqst, so they don't have to fight against it. The Rqst pulse is stored in the asymmetric latch, so that the pull-down

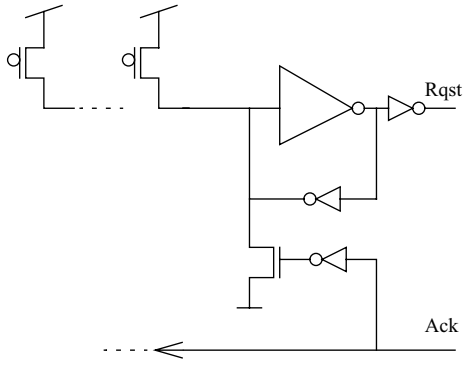

Fig. 8. New communication scheme between the pixels and the arbiter

transistor doesn't have to fight against a strong inverter, but against a weak one.

In short, this communication architecture between the pixels and the arbiter is more robust, it doesn't need a voltage bias, and it is much faster.

B. Image resolution

As was detailed before, we have designed a cluster of 2 x 2 pixels, whose area is the same as the previous version. Consequently, we can build an array 4 times larger of 64 x 64 pixels.

This change in the size of the array made us scale the AER-out block, so that the arbiters receive 64 inputs instead of 32. However, we didn't change the size of the RAM, because for most applications it is not necessary to use kernels larger than 32 x 32.

In this case, the left/right column shifter had to be redesigned, because now there are 32 digital words as inputs, and 64 digital words as outputs.

C. Multikernel capability

The basic idea of the multikernel capability is that we can program several different kernels in the RAM (up to 32), so that each input event will include information about what kernel will be used to process it. This idea will be very useful for multichip multilayer configurations, because with only one chip we will be able to implement the functionalities of several chips, reducing the number of boards in the experimental setup.

To implement this new functionality we had to apply several changes to the following blocks: the configuration registers, the synchronous controller and the left/right column shifter.

- The configuration registers in the previous version were used to set parameters like the address of the array of pixels (when building array of chips) and the size of the programmed kernel. Thereby, the controller had to use the information stored in these registers to decide which rows of the RAM had to be added into the pixels. However, as we wanted to be able to write several kernels in the RAM, we needed to store information about each one of them. So, for each kernel we must specify the horizontal and vertical coordinates inside the RAM, and also the horizontal and vertical distance between the geometrical center and the application center of the kernel (this allows for using non-square kernels that are not applied around its center, as the one shown in Fig. 9). Thus, we need 6 parameters for each one of the 32 possible kernels (and also the same information that we had in the previous version).

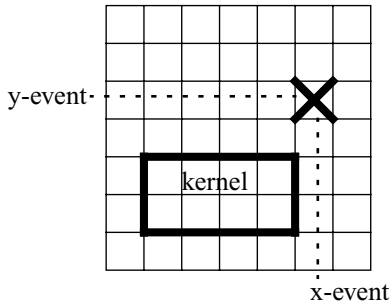

Fig. 9. Example of a non-square kernel with the application center outside of it

- The synchronous controller had to be redesigned, so that it can receive information about the kernel from the input events and process it. In the previous version, as there was only one kernel in the RAM, the controller considered that it was placed at the upper left corner of the RAM, and it had to be square and odd (to make the calculations about the center of the kernel easier). In the new version, when an input event arrives indicating a specific kernel, the controller will have to find out where in the RAM that kernel is written, and activate the corresponding rows of the RAM. So, it will access the information stored at the configuration registers, and generate the necessary output signals.
- The left/right column shifter has to include a new functionality for the implementation of the multikernel capability. In the previous version, as there was only one kernel, the rest of the unused RAM had to be filled with zeros, so that when a row was enabled, the pixels of the selected row that were outside the neighborhood would add zero to the accumulator. However, in the new situation the RAM positions outside of the selected kernel will not always be zero, so we must ensure that these values will not reach the pixels. That's why we include at the input of this block an inhibition circuit that only allows the

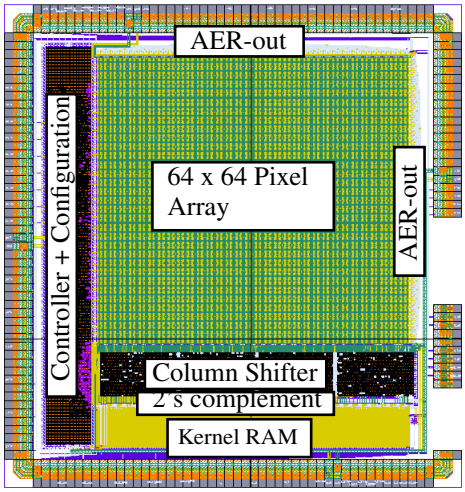

Fig. 10. Layout of the new 64 x 64 fully digital convolution chip

RAM values to reach this block if they receive the order from the controller. They will be seen as zero otherwise.

Fig. 10 shows the layout of the new 64 x 64 fully digital convolution chip with all the described improvements and new functionalities included. The size of the whole chip is 5480 μm x 5780 μm.

5. Conclusions

In this paper, a new version of a convolution chip has been presented, which has been designed and sent out for fabrication.

This chip can be considered as an evolution from the previous works. The most important improvements are: 4 times larger resolution (in this case, 64 x 64) with approximately the same pixel area, reduction of the power consumption, improvement of the communication between the pixels and the arbiter, and the implementation of new functionalities, like the possibility of inhibiting the positive or the negative events, and the multikernel system, that will be very useful to build large multichip test setups.

6. Acknowledgements

This work was supported by EU grant 216777 (NABAB), Spanish grant TEC2006-11730-C03-01 (SAMANTA II) and the local administration from Andalucía grant P06-TIC-01417 (Brain System).

7. References

[1] Luis Alejandro Camuñas-Mesa, Antonio José Acosta-Jiménez, Teresa Serrano-Gotarredona, Bernabé Linares-Barranco, Rafael Serrano-Gotarredona, "Image Processing Architecture Based on a Fully Digital AER Convolution Chip", Proceedings of XXII Conference on Design of Circuits and Integrated Systems, DCIS'07, November 2007, pp. 385-390

[2] G. M. Shepherd, The synaptic organization of the brain, Oxford University Press, 3rd Edition, 1990

[3] M. Sivilotti, "Wiring considerations in analog VLSI systems with application to field-programmable networks", Ph.D. dissertation, Comp. Sci. Div., California Inst. Technol., Pasadena, CA, 1991

[4] T. Serrano-Gotarredona, A. G. Andreou and B. Linares-Barranco, "AER image filtering architecture for vision processing systems", IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process., vol. 46, no. 9, pp. 1064-1071, Sep. 1999

[5] B. Linares-Barranco, T. Serrano-Gotarredona, R. Serrano-Gotarredona, and J. Costas-Santos, "A New Charge-Packet Driven Mismatch-Calibrated Integrate-and-Fire Neuron for processing Positive and Negative Signals in AER based Systems", Circuits and Systems, ISCAS 2004, Proceedings of the 2004 International Symposium, vol. 5, 23-26, pp. V-744 - V-747, 23-26 May 2004

[6] R. Serrano-Gotarredona, T. Serrano-Gotarredona, A. Acosta-Jimenez, and B. Linares-Barranco, "A Neuromorphic Cortical-Layer Microchip for Spike-Based Event Processing Vision Systems", IEEE Trans. Circuits and Systems, Part-I: Regular Papers, vol. 53, No. 12, pp. 2548-2566, December 2006

[7] Luis Alejandro Camuñas-Mesa, Antonio Acosta-Jiménez, Teresa Serrano-Gotarredona, Bernabé Linares-Barranco, "Fully Digital AER Convolution Chip for Vision Processing", IEEE International Symposium Circuits and Systems 2008, ISCAS'08, May 2008, pp. 652-655

[8] Boahen, K. A., "A burst-mode word-serial address-event link-III: analysis and test results", Transactions on Circuits and Systems I: Regular Papers, IEEE, Volume 51, Issue 7, July 2004 Page(s):1292 - 1300